Push Architecture

Candidate Version 2.3 – 16 Mar 2010

Open Mobile Alliance

OMA-AD-Push-V2_3-20100316-C

Contents

Figures

1. Scope (Informative)

The Open Mobile Alliance (OMA) continues the work of the WAP Forum in developing industry-wide specifications for developing enablers and services that operate over wireless communication networks. To enable operators and manufacturers to meet the challenges in advanced services, differentiation, and fast/flexible service creation, OMA defines a set of enablers in transport, session and application layers.

This document outlines the Push Architecture and related specifications, which together specify an enabler for a service to push content to mobile devices.

- **Push Architectural Overview**
  The purpose of this document is to serve as a starting point for anybody wanting to know more about the Push technology, before taking on the other specifications.

- **Push Access Protocol Specification**
  This document specifies the protocol with which a Push Initiator communicates with the PPG. See section 5.3.1.1 for a brief description.

- **Push Proxy Gateway Service Specification**
  This document specifies the Push Proxy Gateway functionality, and how it interacts with the Push Access Protocol and the Push Over-The-Air protocol. See section 5.3.1 for a brief description.

- **Push OTA Protocol Specification**
  This document specifies the protocol with which a PPG communicates with a push-capable client. See section 5.3.2.2 for a brief description.

- **Push OTA Protocol – Cell Broadcast Service (CBS) Adaptation Specification**
  This document specifies the protocol with which a PPG communicates with a Push/CBS-capable client. See section 5.3.2.2 for a brief description.

- **Push Client-Application Interface Specificiation**
  This document specifies the optional interface between a Push Client and applications, via which the application can register for push services and receive push content. See section 5.3.2.3 for a brief description.

- **Push Message Specification**
  This document specifies end-to-end properties of a push message.

- **Service Indication Specification**
  This document specifies a content type used for notifying users they have new information waiting on a server. See section 5.3.2.1.2.1 for a brief description.

- **Service Loading Specification**
  This document specifies a content type that instructs the client to automatically load a URI. See section 5.3.2.1.2.2 for a brief description.

- **Cache Operation Specification**
  This document specifies a content type that instructs the client to invalidate cached resources. See section 5.3.2.1.2.3 for a brief description.

2. References

2.1 Normative References

[3GPP-CBS] "Technical Realization of Cell Boradcast Service (CBS)", 3GPP TS23.041, url:http://www.3gpp.org/

[23.246] "Multimedia Broadcast/Multicast Service (MBMS); Architecture and functional description", 3GPP TS 23.246, url:http://www.3gpp.org/

[26.346] "Multimedia Broadcast/Multicast Service (MBMS); Protocols and codecs", 3GPP TS 26.346, url:http://www.3gpp.org/

[DMSTDOBJ] "OMA Device Management Standardized Objects, Version 1.2". Open Mobile Alliance™ OMA-TS-DM-StdObj-V1_2. URL:http://www.openmobilealliance.org

[OMA-BCAST-AD] "Mobile Broadcast Services Architecture", Open Mobile Alliance™, OMA-AD-BCAST-V1_0, URL:http://www.openmobilealliance.org/

[OMNA] "OMA Naming Authority". Open Mobile Alliance™. URL: http://www.openmobilealliance.org/

[PushCBS] "Push Over the Air Technical Specification – CBS Adaptation" . Open Mobile Alliance™. OMA-TS-Push_CBS_Adaptation-V1_0. URL:http://www.openmobilealliance.org/

[PushCO] "Cache Operation" WAP-175-CacheOp. Open Mobile Alliance™. URL: http://www.openmobilealliance.org/

[PushMO] "Push Management Object". Open Mobile Alliance™. OMA-TS-Push_MO-V1_1. URL: http://www.openmobilealliance.org/

[PushMsg] "Push Message Specification". Open Mobile Alliance™. OMA-TS-Push_Message-V2_3. URL: http://www.openmobilealliance.org/

[PushOTA] "Push OTA Protocol Specification". Open Mobile Alliance™. OMA-TS-PushOTA-V2_3. URL: http://www.openmobilealliance.org/

[PushPAP] "Push Access Protocol Specification". Open Mobile Alliance™. OMA-TS-PAP-V2_3. URL:http://www.openmobilealliance.org/

[PushPPG] "Push Proxy Gateway Service Specification". Open Mobile Alliance™. OMA-TS-PPGService-V2_3. URL:http://www.openmobilealliance.org/

[PushSI] "Service Indication". Open Mobile Alliance™. WAP-167-ServiceInd URL:http://www.openmobilealliance.org/

[PushSL] "Service Load". Open Mobile Alliance™. WAP-168-ServiceLoad URL:http://www.openmobilealliance.org/

[PushSIP] "SIP Push". Open Mobile Alliance™. OMA-TS-SIP_Push-V1_0. URL:http://www.openmobilealliance.org/

[PushSIP-AD] "Push using SIP Architecture", Open Mobile Alliance™. OMA-AD-SIP_Push-V1_0, URL:http://www.openmobilealliance.org/

[Push-CAI] "Push Client-Application Interface", Open Mobile Alliance™. OMA-TS-SIP-PushCAI-V1_1, URL:http://www.openmobilealliance.org/

[MBMS] Multimedia Broadcast/Multicast Service (MBMS); Protocols and codecs; 3GPP TS 26.346, URL: http://www.3gpp.org/

[RFC1951] "DEFLATE Compressed Data Format Specification version 1.3". P. Deutsch. May 1996. http://www.ietf.org/rfc/rfc1951.txt

[RFC1952] IETF RFC 1952 (May 1996): "GZIP file format specification version 4.3", P. Deutsch.
http://www.ietf.org/rfc/rfc1952.txt

[RFC2387] "The MIME Multipart/related content type". E. Levinson. August 1998.
URL:http://www.ietf.org/rfc/rfc2387.txt

[RFC2616] "Hypertext Transfer Protocol – HTTP/1.1". R. Fielding et al. June 1999.
URL:http://www.ietf.org/rfc/rfc2616.txt

[RFC2617] "HTTP Authentication: Basic and Digest Access Authentication". J. Franks et al. June 1999.
URL:http://www.ietf.org/rfc/rfc2617.txt

[RFC3261] "SIP: Session Initiation Protocol". J. Rosenberg et al. June 2002. URL:http://www.ietf.org/rfc/rfc3261.txt

[RFC 3452] "Forward Error Correction (FEC) Building Block", M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, J. Crowcroft December 2002, URL:http://www.ietf.org/rfc/rfc3452.txt

[RFC4975] Campbell, B., Ed., Mahy, R., Ed., and C. Jennings, Ed., "The Message Session Relay Protocol (MSRP)", RFC 4975, September 2007.
URL: http://www.ietf.org/rfc/rfc4975.txt

[UAProf] "OMA User Agent Profile".OMA-UAProf-v2_0. URL:http://www.openmobilealliance.org/

[WAE] "OMA Wireless Application Environment Specification".OMA-WAP-TS-WAESpec-V2_3.
URL:http://www.openmobilealliance.org/

[WAPTLS] "WAP TLS Profile and Tunnelling". WAP-219-TLS. URL:http://www.openmobilealliance.org/

[WBXML] "Wireless Binary XML". WAP-192-WBXML. URL:http://www.openmobilealliance.org/

[W-HTTP] "Wireless Profiled HTTP". Open Mobile Alliance™. WAP-229-HTTP,
URL:http://www.openmobilealliance.org/

[WSP] "Wireless Session Protocol". Open Mobile Alliance™. WAP-230-WSP
URL:http://www.openmobilealliance.org/

[WTLS] "Wireless Transport Layer Security Protocol". Open Mobile Alliance™. WAP-261-WTLS.
URL:http://www.openmobilealliance.org/

[XML] "Extensible Markup Language (XML) 1.0 (Second Edition)". T. Bray, et al, October 2000. URL:
http://www.w3.org/TR/REC-xml

2.2 Informative References

None.

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.”

3.2 Definitions

Application	A value-added data service provided to a client. The application may utilise both push and pull data transfer to deliver content
Application-Level Addressing	The ability to address push content between a particular user agent on a WAP client and push initiator on a server
Bearer Network	A network used to carry the messages of a transport-layer protocol between physical devices.
Client	A device, user agent, or other entity that acts as the receiver of a service [OMADICT]
Contact Point	Address information that describes how to reach a push proxy gateway, including transport protocol addres and port of the push proxy gateway.
Content	Digitized work that is processed, stored, or transmitted. It includes such things as text, presentation, audio, images,video, executable files, etc. Content may have properties such as media type, mime type, etc [OMADICT]
Content Encoding	When used as a verb, content encoding indicates the act of converting a data object from one format to another. Typically the resulting format requires less physical space than the original, is easier to process or store, and/or is encrypted. When used as a noun, content encoding specifies a particular format or encoding standard or process.
Content Format	Actual representation of content.
Device	Equipment which is normally used by users for communications and related activities. The definition can be extended to cover remote monitoring applications where there is no user present, but the communications to and from the remote monitor use the same communications channels as when used by users [OMADICT]
End-user	An individual who uses services and content [OMADICT]
Extensible Markup Language	See “XML”
Media Type	A MIME media type or an identifier for a given data type [OMADICT]
Multicast Message	A push message containing a single OTA client address which implicitly specifies more than OTA client address.
Point-to-Multipoint Push	Push content delivery to a group of users through the OTA-PTM Push-OTA protocol variant.
Push Access Protocol	A protocol used for conveying content that should be pushed to a client, and push related control information, between a Push Initiator and a Push Proxy/Gateway.
Push Application	A value-added data service provided to a client. The application may utilise both push and pull to deliver content.
Push Framework	The entire push system. The push framework encompasses the protocols, service interfaces, and software entities that provide the means to push data to user agents in the client.
Push Initiator	An entity or service that intiates Push content delivery to Push clients [OMADICT]
Push OTA Protocol	A protocol used for conveying content between a Push Proxy/Gateway and a certain user agent on a client.

Push Proxy Gateway	A gateway acting as a Push proxy for Push Initiators, providing over-the-air Push message delivery services to Push clients [OMADICT]
Push Session	A WSP session that is capable of conducting push operations. Multiple bearer networks may be used over the life of a single push session.
Server	An entity that provides resources to clients in response to requests [OMADICT]
SIP Push	Push content delivery to a specific user via the OTA-SIP Push-OTA protocol variant.
Terminal equipment	Equipment that provides the functions necessary for the operation of the access protocols by the user (source: GSM 01.04).
User	An entity which uses services. Example: a person using a device as a portable telephone [OMADICT]
User agent	Any software or device that acts on behalf of a user, interacting with other entities and processing resources [OMADICT].
WAP Push	Push content delivery to a specific user via the WAP1 (OTA-WSP) or WAP2 (OTA-HTTP) Push-OTA protocol variants.
XML	The Extensible Markup Language is a World Wide Web Consortium (W3C) standard for Internet markup language, of which WML is one such language [OMADICT]

3.3 Abbreviations

CPI	Capability and Preference Information
GRUU	Globally Routable User Agent (UA) URIs
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
MIME	Multipurpose Internet Mail Extensions
MSISDN	Mobile Station International Subscriber Directory Number
OMNA	Open Mobile Alliance Naming Authority
OTA	Over The Air
OTA-HTTP	(Push) OTA over HTTP
OTA-PTM	(Push) OTA over Point-to-Multipoint protocols
OTA-WSP	(Push) OTA over WSP
PAP	Push Access Protocol
PI	Push Initiator
PO-TCP	PPG Originated TCP connection establishment method
PPG	Push Proxy Gateway
PTM-Push	Point-to-Multipoint Push
QoS	Quality of Service
SGML	Standard Generalized Markup Language
SI	Service Indication
SIA	Session Initiation Application
SIP	Session Initiation Protocol
SIR	Session Initiation Request
SL	Service Loading
SMS	Short Message Service

SSL	Secure Socket Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TO-TCP	Terminal Originated TCP connection establishment method
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WAP	Wireless Application Protocol
WBXML	WAP Binary XML
WSP	Wireless Session Protocol
WTLS	Wireless Transport Layer Security
XML	Extensible Mark-up Language

4. Introduction (Informative)

In the "normal" client/server model, a *client* requests a service or information from a *server*, which then responds in transmitting information to the client. This is known as “pull” technology: the client “pulls” information from the server. Browsing the World Wide Web is a typical example of pull technology, where a user enters a URL (the request) that is sent to a server, and the server answers by sending a Web page (the response) to the user.

In contrast to this, there is also “push” technology, which is also based on the client/server model, but where there is no explicit request from the client before the server transmits its content. The Push framework introduces a means to transmit information to a device without a user request.

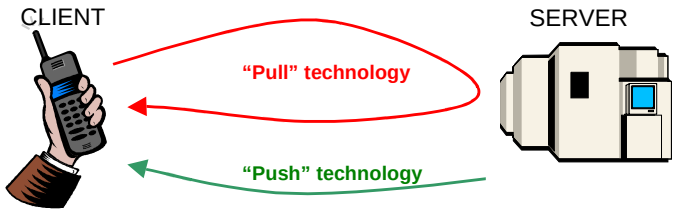

Figure 1 - Comparison of pull vs. push technology

Another way to say this is that whereas “pull” transactions of information are always initiated from the client, “push” transactions are server-initiated.

The OMA Push architectural model enables server-to-client content delivery via various communication methods (protocols), interconnection types (point-to-point, point-to-multipoint), and transport bearers (underlying protocols and bearer networks).

OMA Push enables a client/server relationship between the two main entities in the OMA Push architecture: a *Push Proxy Gateway (PPG)* which act as the Push Server, and a *Push Client*. This client/server relationship supports content delivery to one or more Push Clients via discrete point-to-point connections (unicast), and delivery a group of Push Clients via shared point-to-multipoint connections (multicast/broadcast). Collectively, the set of OMA Push-defined protocols (and binding to underlying protocols and bearers) enabling client/server interaction is referred to as *OMA Push Over-the-Air (Push-OTA)*. Note the name Push-OTA is based upon the historical focus of OMA Push on mobile data services, but the protocol is also applicable for any IP-based network environment.

As indicated by its name, the PPG has various roles including acting as a proxy for push operations initiated by external applications, and as a gateway for converting external application/interface operations for delivery via a Push-OTA variant. PPG functions in support of these roles include Push Client registration, Push Client address translation, push content transformation, and management of push operations as transactions in a store-and-forward mode of operation.

The set of protocols and interconnection types supported in OMA Push are intended to ensure the feasibility of delivering Push-based services across the wide variety of transport bearers via which the PPG and Push Client may be interconnected. Within this broader set of options, the OMA Push functions of the PPG and Push Client may be only partially implemented or deployed, according to the needs/capabilities of a specific service environment. For example, an application server may act as a Push Server, directly supporting one or more Push-OTA protocol variants, without implementing the broader Push service capabilities of a PPG.

Through their external interfaces, the PPG and Push Client may enable Push-based service delivery on behalf of external entities: a *Push Initiator (PI)* and a *Client Application*. From the end-to-end perspective, a push operation is accomplished by allowing a PI to transmit *push content* and *delivery instructions* to a PPG, which delivers the push content to the Push Client according to the delivery instructions. The Push Client subsequently delivers the push content to Client Application.

A PI can be any server-based application that initiates push operations via a PPG, using the *OMA Push Access Protocol* (PAP). The PPG uses a Push-OTA protocol variant to deliver the push content to the Push Client.

Client applications may be OMA enabler user agents (e.g. browsers, multimedia messaging clients, instant messaging clients, etc) or other device-resident applications that are supported by the Push Client.

Figure 2 illustrates the push framework:

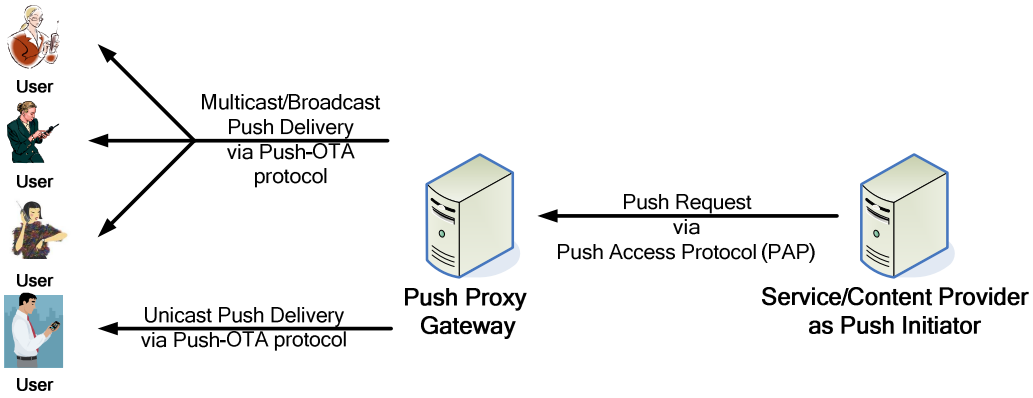

Figure 2: The Push Architectual Model

PAP is based on standard Internet protocols; XML is used to express the delivery instructions, and the push content can be any MIME media type. These standards help make Push flexible and extensible.

The Push-**OTA** protocol provides both "connectionless" and "connection-oriented" services. Connection-oriented service refers to a service context in which the Push Client has established a specific transport layer "connection" with the PPG, for reception of Push-based services. Connection-oriented service is supported via the WAP1 Wireless Session Protocol (WSP) as the OTA-WSP Push protocol, via the WAP2 HTTP-based OTA-HTTP Push protocol, and via the SIP-based OTA-SIP Push protocol. Connectionless service does not depend upon a pre-established/specific connection between the Push Client and PPG, and is supported via OTA-WSP, OTA-SIP, and OTA-PTM.

OTA-PTM is the protocol variant used in Point-to-Multipoint Push (PTM-Push), referring to Push operation across point-to-multipoint bearers, which in Push 2.3 includes MBMS, CBS, and BCAST. OTA-PTM is considered to support connectionless service. While the Push Client actively listens for Push events over the OTA-PTM bearers, there is no specific dialog created between the Push Client and PPG.

An important adjunct to connection-oriented service is the Session Initiation Application (SIA), which is further described in section 5.3.2.2.5.

Figure 2 illustrates the PI and the PPG as separate entities, although the PI and the PPG may be co-located depending upon various deployment-specific needs. Figure 3 below provides a more detailed end-to-end view of the service environment supported by the OMA Push enabler.

The following OMA Push-defined aspects of Figure 3 are described in further detail in this specification and the technical specification of the OMA Push enabler:

- Architectural entities: Push Client and Push Proxy Gateway
- Interfaces
  - Push Over-the-Air protocol variants
    - OTA-WSP: point-to-point delivery based upon WAP1 transport protocols
    - OTA-HTTP: point-to-point delivery based upon WAP2 transport protocols
    - OTA-SIP: point-to-point delivery based upon SIP transport protocols
    - OTA-PTM: point-to-multipoint delivery based upon MBMS, OMA BCAST, and Cell Broadcast Service (CBS) as transport protocols

        - Push Access Protocol, via which server-based applications initiate content delivery through the OMA Push enabler
        - Client-Application Interface, via which client-based applications access Push-based services provided by the OMA Push enabler
    - Support provided by external entities with specific roles as defined by the OMA Push enabler, including
        - OMA User Agent Profile documents, as obtained from web servers, defining static capabilities of Push Clients and Client Applications
        - OMA Device Profile Evolution (DPE) servers, providing access to static and dynamic capabilities of Push Clients and Client Applications
        - OMA Device Management (DM), providing Management Object (MO) support for OMA Push, e.g. for managing the Push access control whitelist
    - Support provided to external entities via specific features defined by the OMA Push enabler, including the identification of OMA-standardized enabler clients by assigned Push Application ID's, and the use of OMA Push-defined content types
        - OMA Browsing user agent (the device's default web browser), identified by the OMA Push enabler as the "WML user agent" (meaning the handler of Web browsing related services)
        - Other OMA-standardized enabler clients, e.g. user agents supporting the OMA-defined enablers Multimedia Messaging Service (MMS), Email, Instant Messaging (IM), Data Sync, Location, Device Management, Download, etc

The following external entities provide functions and interfaces used by the Push Client and/or Push Server, for the adaptation of Push-OTA protocol variants:

- On the network side
    - Short Message Service Centers (SMSC) via which OTA-WSP/SMS is provided
    - Cell Broadcast Centers (CBC), via which OTA-PTM/CBS is provided
    - SIP/IP Core networks, via which OTA-SIP is provided
    - MBMS Broadcast Multicast Service Centers (BM-SC), via which OTA-PTM/MBMS is provided
    - OMA BCAST Service Distribution/Adaptation, via which OTA-PTM/BCAST is provided
- On the client side
    - SMS Clients, via which OTA-WSP/SMS is delivered
    - OMA BCAST Clients, which OTA-PTM/BCAST is delivered
    - MBMS Clients, via which OTA-PTM/MBMS is delivered

The following external entities are examples of those with general supporting roles as typically represented in OMA Push deployments, e.g. for service personalization and contextualization:

- Location servers, providing useful information about the target client location, e.g. for bearer selection and queueing
- Group management servers, providing information of use in group-based Push services, e.g. association of a group identity to multicast address, or identity of specific group members for use in unicast delivery
- Subscriber profile servers, providing information of use in Push-based service personalization, e.g. subscriptions, delivery preferences, subscriber-associated client identities

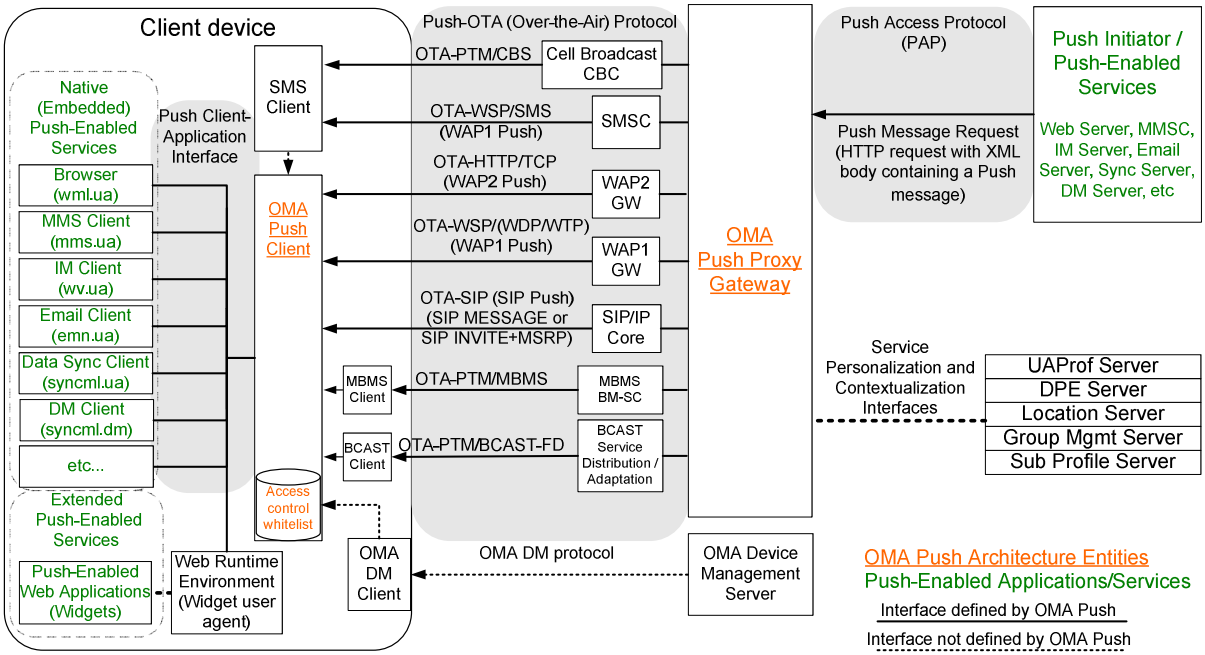

Figure 3 End-to-End Push Framework

4.1 Version 2.2 Functionality

This enabler release continues on the work of the OMA in the area of Push and is an extension of the Push 2.1 Enabler release [Push2.1], defining push security mechanisms and OTA-SIP as a new Push-OTA protocol variant. An aspect of the defined push security mechanisms depend on device management object extension defined in [PushMO] which depends on the OMA Device Management Enabler .

In addition this enabler release definition defines a minimum level of conformance for segmentation and re-assembly for SMS based Push, as well as push initiator guidelines on the most efficient way to use this form of push delivery mechanism.

4.2 Version 2.3 Functionality

This enabler release is an extension of the Push 2.2 Enabler release [Push2.2], and is referred to as Push 2.3 Point-to-Multipoint Push (PTM-Push). PTM-Push adds multipoint distribution methods to complement the existing point-to-point methods, enabling Push content delivery to a large number of clients simultaneously via network bearers supporting multicast and broadcast operation, e.g. MBMS, Cell Broadcast Service (CBS), and OMA BCAST.

PTM-Push defines push security mechanisms for use multipoint service contexts, and OTA-PTM as a new Push-OTA protocol variant. Aspects of the defined push security mechanisms and Push Client configuration depend on a device management object extension defined in [PushMO] which depends on the OMA Device Management Enabler [DMSTDOBJ].

5. Architectural Model

5.1 Dependencies

The WAP1 Push-OTA protocol variant OTA-WSP is dependent upon the WAP1 user agent and protocol specifications [WSP], [WBXML], and [WTLS].

The WAP2 Push-OTA protocol variant OTA-HTTP is dependent upon the WAP2 protocol specifications [W-HTTP] and [WAPTLS].

The OTA-SIP protocol variant is dependent upon the SIP and SIP-related protocol specifications described in [PushSIP].

The OTA-PTM protocol variant is dependent upon the protocol specifications described in [MBMS], [OMA-BCAST-AD], and [3GPP-CBS].

The user agent environment specification [WAE] and user agent profile specification [UAProf] are dependencies on functions of the Push Client and Push Proxy Gateway entities in the OMA Push architecture.

5.2 Architectural Diagram

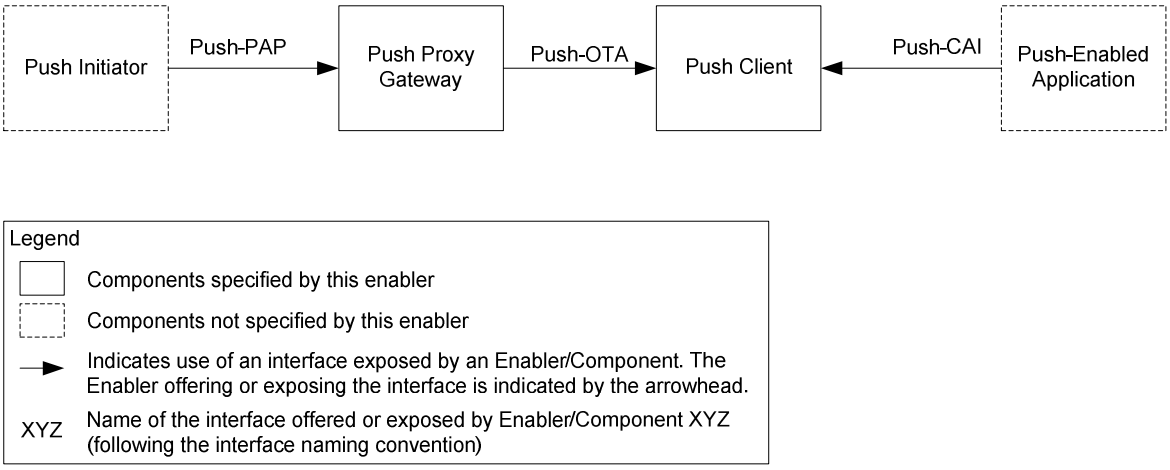

Figure 4 - Architecture Diagram

The interfaces shown in Figure 4 include:

- The Push-OTA interface, via which the PPG and Push Client interact using one of the Push-OTA protocol variants (see section5.3.2.2)
- The Push-PAP interface, via which the PPG exposes Push services to the Push Initiator (see section 5.3.2.1)
- The Push-CAI interface, via which the Push Client exposes Push services to Push-enabled applications (see section 5.3.2.3)

5.3 Functional Components and Interfaces/reference points definition

5.3.1 Functional Components

5.3.1.1 The Push Proxy Gateway

The Push Proxy Gateway (*PPG*) is the entity that does most of the work in the Push framework. Its responsibilities include acting as an access point for content pushes from Push Intiators to Push Clients, and everything associated therewith (authentication, address resolution, etc).

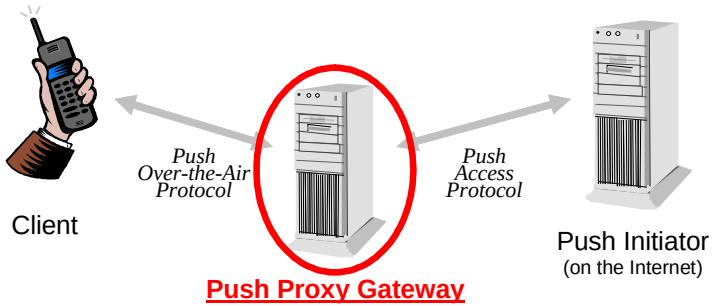

Figure 5 - PPG highlighted

The PPG may implement access-control policies about who is able to push content and who is not, and under which circumstances, etc.

5.3.1.1.1 Services Overview

The PPG provides the Push framework with several services. It is the initial contact point for content pushed to the Push Client. The PPG may perform the following:

- PI identification and authentication; access control
- Parsing of and error detection in push content and control information
- Push Client discovery services (including Push Client capabilities)
- Address resolution of push recipient
- Binary encoding and compilation of certain content types, or general compression, to improve efficiency
- Protocol conversion

5.3.1.1.2 Message Acceptance and Rejection

The PPG accepts content from the PI using the Push Access Protocol (see section5.3.2.1). This content is divided into several parts using a multipart/related content type where the first part contains control information for the PPG itself. Such information includes recipient address(es), delivery time constraints, Quality of Service (QoS) information, notification requests, etc.

The PPG will acknowledge successful (or report unsuccessful) parsing of the control information, and may in addition report debug information about the push content itself.

5.3.1.1.3 Message Handling Service

Once the content has been accepted for delivery, the PPG attempts to deliver the content to the target Push Client (or group of Push Clients, for OTA-PTM) using the Push Over-The-Air protocol (see section 5.3.2.2). The PPG may attempt to deliver the content until a timeout expires. This timeout may be set by the PI and/or policies of the PPG operator.

The PPG may also send a notification to the PI when the final status of the push submission (delivered, cancelled, expired, etc.) has been reached, if the PI so requests. Hence, the service is asynchronous from the PI's point of view (the PI need not wait on-line for the PPG to complete its delivery).

5.3.1.1.4 Encoding, Compilation, and Compression

The PPG may encode content types (e.g. WML 1.2 and SI) into their binary counterparts [WBXML]. This textual-to-binary translation would take place before delivery over-the-air. Other content types, unknown to the PPG, may be forwarded as received unless it is known that the Push Client does not support them.

When OTA-HTTP (see section 5.3.2.2.4) or OTA-SIP (see section 5.3.2.2.3) is used, the PPG may use deflate encoding [RFC1951] (i.e. general compression) to obtain better over-the-air efficiency also for content types that do not have an associated binary content type.

When OTA-PTM (see section 5.3.2.2.4) is used, the PPG may use GZip [RFC 1952] to encode files for transport. Also the service data may be protected against packet loss using Forward Error Corretion [RFC3452]. Note, other reliability schemes e.g. File Repair using point to point bearers are also defined in [MBMS].

5.3.1.1.5 Multicast, Broadcast and Aliasing Considerations

The nature of multicast and broadcast services enables Push delivery to groups of Push clients. The association of users as a group to a Push-enabled service may be supported by the PPG through implementation-specific address aliasing schemes, e,g., via special addresses used to target Push content to groups of users via OTA-PTM. PTM-Push provides for identifying the Push target by name, which can be used by the PPG for associating a Push request to such an alias group.

When receiving a Push message indicated to be delivered using an applicable PTM-Push bearer, the PPG will ensure at least one bearer among those bearers is to be used. A specific PTM-Push bearer can be indicated by the PI in the PAP message, e.g. indicated by a named user group, a broadcast/multicast address, targeted location, or network identifier for PTM-Push. For the specific bearer, the PPG needs to ensure whether the bearer is available, e.g. based upon the available network resources. If yes, the PPG will select the bearer as indicated, further adapting the Push message to it, and deliver it to the Push Client by using selected PTM-Push bearer. If the bearer indicated is not available or no specific bearer is indicated, if possible the PPG selects an available bearer, e.g. based on group or network information. PPG may respond with the selected bearer to PI if requested.

While the manangement of Push target alias groups is out of scope for PTM-Push, the PPG does associate PTM-Push target addresses to PTM-Push adaptation requirements, i.e. bearer configuration and OTA-PTM protocol options related to the specific PTM-Push bearer such as:

- Type of PTM-Push bearer to use, e.g. for a group address that is associated with a specific PTM-Push bearer.
- OTA addressing for the PTM-Push bearer, which may include scoping the Push request to a specific service area, e.g. geographical area

Parameter negotiation for the selected bearer may be required to fulfil the adaptation requirements of the specific multicast/broadcast bearer.

5.3.1.1.6 Client Capabilities Query Service

A PI may query the PPG for a specific Push Client's capabilities and preferences to aid in creating better-formatted content. The capability and preference information returned by the PPG is formatted as a UAProf document [UAProf]. This feature is optional in the PPG.

5.3.1.1.7 Reference

For more information, see [PushPPGPushPPG].

5.3.1.2 The Push Client

The Push Client is the entity that receives Push content from the PPG. Its responsibilities include acting as a service access point for OMA enabler user agents or applications that directly use Push services.

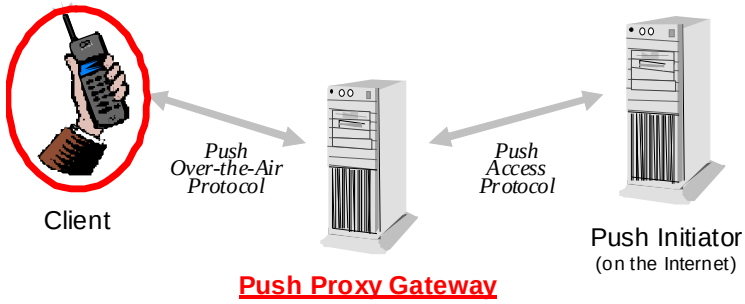

Figure 6 – Push Client highlighted

Push Clients may include support for a specific set of OMA enabler user agents (e.g. browser), and may additionally support the dynamic registration of new user agents via the Push Client-Application Interface[Push-CAI]. As part of supporting the Push-CAI, Push Clients may implement access-control policies about which user-agent/applications are able to access Push services.

Push Clients have several roles in the Push enabler:

- Connection/registration with a PPG as needed, e.g. for OTA-WSP service over IP bearers, OTA-HTTP service, and OTA-SIP service
- Push content reception from the PPG via the various Push-OTA variants, including handling Push Security
- Push content delivery to applicable user agents and applications, via Push-CAI or other unspecified device-internal interfaces

5.3.2 Interfaces

5.3.2.1 Push Access Protocol Interface

The Push Access Protocol (*PAP*) is the means by which a PI pushes content to a mobile network, addressing its PPG.

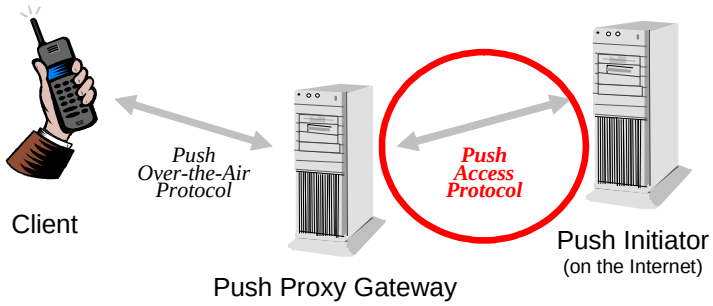

Figure 7 - PAP highlighted

PAP was designed to be independent of the underlying transport; it can be transported over virtually any protocol that allows MIME types to be transported over the Internet. HTTP is the first protocol to be specified as a transport protocol for PAP, other protocols (e.g. SMTP) may be added in the future.

When HTTP is used as a transport protocol for PAP, the HTTP POST request method and its response are used to transport the information. The HTTP response always contains result code 202 (“accepted for processing”) when the HTTP transaction itself succeeds; the response PAP document may contain a PAP error though. See [RFC2616] for more information about HTTP/1.1.

PAP is used to carry push related control information that is used by the PPG. This information is expressed using XML [XML]. When, for example, a new message is submitted to the PPG the control information and the push content are carried in a MIME multipart/related [RFC2387] entity. This implies that a single (MIME) entity is conveyed independent of the type of operation.

5.3.2.1.1 PAP Operations

The PAP supports the following operations:

- Push Submission (PI to PPG)
- Result Notification (PPG to PI)
- Push Cancellation (PI to PPG)
- Push Replacement (PI to PPG)
- Status Query (PI to PPG)
- Client Capabilities Query (PI to PPG)

These operations are described in the following subsections. For more information, see [PushPAP].

5.3.2.1.1.1. Push Submission

The Push message contains three entities: a control entity, a content entity, and optionally a capability entity. These are bundled together in a multipart/related message, which is sent from the PI to the PPG.

The control entity is an XML document that contains delivery instructions destined for the PPG, whereas the content entity is destined for the Push Client. The PPG may or may not convert the content entity into a more bandwidth-optimised form before forwarding it to the Push Client (see section 5.3.1.1.4).

The optional capability entity contains the Push Client capabilities that the message was formatted according to, in UAPROF [UAProf] format. The PI may include this entity to indicate what it *assumes* the Push Client capabilities are. The PPG may reject the message if the assumed capabilities do not match those known by the PPG.

5.3.2.1.1.2. Result Notification

If the PI has requested information about the final outcome of the delivery, the Result Notification message is transmitted from the PPG to the URI specified by the PI. It is an XML entity indicating successful or unsuccessful (Push Client not reachable, timeout, etc.) delivery.

One key feature of the Push Framework is the possibility for a PI to rely on the response from the PPG; a confirmed push is confirmed by the device when (and only when) the target application has taken responsibility for the pushed content. If it cannot take that responsibility, it must abort the operation, and the PI will know that the content never reached its destination.

5.3.2.1.1.3. Push Cancellation

This is an XML entity transmitted from the PI to the PPG, requesting cancellation of a previously submitted message. The PPG responds with an XML entity indicating if the cancellation operation was successful.

5.3.2.1.1.4. Push Replacement

The push submission operation described in section 5.3.2.1.1.1 may – if the PI so requests – cause a previously submitted message to be replaced. It is possible to specify if the new message only should be sent to those recipients whom have not received the original message, or if the new message should be sent to all recipients. In either case, the original message is cancelled for those recipients to whom the original message has not been delivered.

5.3.2.1.1.5. Status Query

This is an XML entity transmitted from the PI to the PPG, requesting the status of a previously submitted message. The PPG responds with an XML entity containing the current status.

5.3.2.1.1.6. Client Capabilities Query

This is an XML entity transmitted from the PI to the PPG, requesting the capabilities of a particular device on the network (see also *Client Capabilities Query*, section 5.3.1.1.6). The PPG responds with a multipart/related entity containing two parts, where the first part contains the result of the request, and the second part contains the capabilities of the device formatted according to the User Agent Profile vocabulary [UAProf].

5.3.2.1.2 Push Specific Media Types

The Push framework allows any MIME media type to be delivered between the PI and the Push Client. The media types described in this section have been created to add capabilities not already provided by existing MIME types. Other media types with push specific semantics have been defined by the OMA to meet the needs for specific applications (e.g. MMS and Provisioning).

Note: If push specific semantics are neither defined for the media type itself, nor for the user agent receiving a certain media type, such media types are placed in the cache memory or discarded when received via push (this applies to e.g. WML). For more information, see [WAE].

5.3.2.1.2.1. Service Indication

The *Service Indication* (SI) media type provides the ability to send notifications to end-users in an asynchronous manner. Such notifications may, for example, be about new e-mails, changes in stock price, news headlines, advertising, reminders of e.g. low prepaid balance, etc.

In its most basic form, an SI contains a short message and a URI indicating a service. The message is presented to the end-user upon reception, and the user is given the choice to either start the service indicated by the URI immediately, or postpone the SI for later handling. If the SI is postponed, the Push Client stores it and the end-user is given the possibility to act upon it at a later point of time.

In addition to the basic functionality described above, SI allows the PI to control the following:

- The level of user-intrusiveness (assign an SI a certain priority)
- Replacement (replacement of an older SI with a new one upon reception)
- Deletion (delete an already received SI by sending a "delete" SI)
- Expiration (assign an expiration time to an SI after which it will be expired)

SI is the only media type among those described in this section that is mandatory in Push Clients implementing push.

For more information, see [PushSI].

5.3.2.1.2.2. Service Loading

In contrast to SI, *Service Loading* (SL) does not imply any user involvement. An SL conveys an URI that points to some content that is loaded by the Push Client without end-user confirmation, and an instruction whether the content should be executed/rendered or placed in the cache memory. If the content should be executed/rendered, the PI can control the level of user-intrusiveness.

For more information, see [PushSI].

5.3.2.1.2.3. Cache Operation

The *Cache Operation* (CO) media type provides a means for invalidating specific objects, or all objects sharing the same URI prefix, stored in the Push Client's cache memory. This feature is useful in situations when the cached content's expiration time cannot be determined beforehand (e.g. a view of a mailbox) and the content changes (e.g. new emails arrive) more often than the user accesses it.

For more information, see [PushCO].

5.3.2.1.3 Addressing

Push addressing occurs on Push Client and application levels.

5.3.2.1.3.1. Push Client Addressing

Between the PI and PPG, the intended recipient(s) of a Push message, also referred to as the Push target, are identified using addresses with specific meaning to the PPG, or with arbitrary identifiers.

The PI uses the Push Client address to instruct the PPG which Push Client the pushed message is intended for. The [PushPPG] specification introduces an addressing scheme that allows:

- **User-defined identifiers**
  An arbitrary text string (e.g. an email address) is used to identify the Push Client. The address may be associated with a specific device/user, or a set of devices/users (e.g. a named user group or targeted location for PTM-Push). The PPG is responsible for translating the string into an address format understood by the mobile network.
  Examples from [PushPPGPushPPG*]:*
  ```
  WAPPUSH=john.doe%40oma.org/TYPE=USER@ppg.carrier.com
  ; user-defined identifier for john.doe@oma.org
  WAPPUSH=+155519990730/TYPE=USER@ppg.carrier.com
  ; user-defined identifier that looks like a phone number
  ```
- **Device addresses**
  An address understood by the mobile network, e.g. MSISDN (SMS etc.), IP address (GPRS etc.), or SIP address. The address may identify a specific device/user, or a set of devices/users (e.g. a broadcast/multicast address for PTM-Push).
  *Examples from [*PushPPGPushPPG*]:*
  ```
  WAPPUSH=+155519990730/TYPE=PLMN@ppg.carrier.com
  ; device address for a phone number of some wireless network
  WAPPUSH=195.153.199.30/TYPE=IPv4@ppg.carrier.com
  ; device address for an IP v4 address
  WAPPUSH=”sip%3Abob%40atlanta.com”/TYPE=URI@ppg.carrier.com
  ; device address for a SIP public address
  WAPPUSH=”sip%3Abob%40atlanta.com%3Bgr=urn%3Auuid%3Af81d4fae-7dec-11d0-a765-
  00a0c91e6bf6”/TYPE=URI@ppg.carrier.com
  ; device address for a SIP public GRUU
  ```

The `TYPE` switch indicates the type of address (user-defined or device address including type of address), and the `ppg.carrier.com` part is the Internet host name of the PPG. For more information, see [PushPPG].

5.3.2.1.3.2. Application-Level Addressing

Pushed content always targets a specific user agent (or more general, a specific application) on the device. An application identifier, which is a URI or a numeric value registered with [OMNA], identifies a user agent. The PI includes the application identifier in a push message by including the `X-Wap-Application-Id` header defined in [PushMsg]. This header is also conveyed to the Push Client when the PPG delivers the message, allowing the Push Client to dispatch the message to the intended user agent.

The PI may use a numeric application identifier as described in section 5.3.2.2.6.1. The PI may also indicate an identifier that is not registered. The latter is discouraged with deployed applications because of the possibility of collisions. It is mainly intended for experimental user agents that have not yet been publicly deployed.

5.3.2.2 Push Over-the-Air Interface

The Push Over-The-Air (OTA) protocol is the part of the Push Framework that is responsible for transporting content from the PPG to the Push Client and its user agents. It is designed to run on top of HTTP (OTA-HTTP), WSP (OTA-WSP), SIP (OTA-SIP), point-to-multipoint multipoint bearers (OTA-PTM).

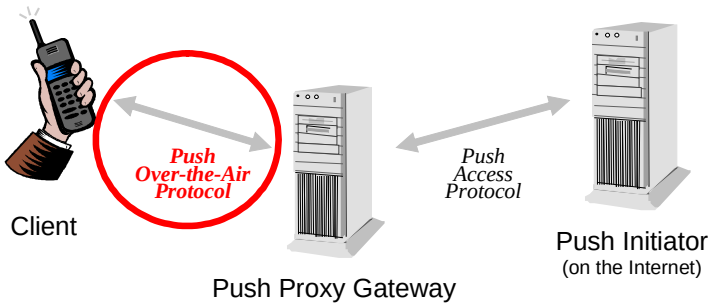

Figure 8 - OTA highlighted

The Push-OTA protocol variants and features are described in the following subsections. For more information, see [PushOTA].

5.3.2.2.1 OTA-WSP

The OTA-WSP protocol variant is architecturally a thin protocol layer on top of WSP, and may hence be used with any bearer addressed by OMA. OTA-WSP utilises the features provided by WSP (see [WSP] for details), and extends those to address push specific needs; basically by introducing new service primitives and extending existing ones with new header fields. For example, OTA-WSP relies upon WSP's capability negotiation feature (possibly using UAProf), and it provides both connectionless (unconfirmed push) and connection-oriented (unconfirmed and confirmed push) services.

5.3.2.2.2 OTA-HTTP

This protocol variant utilises HTTP for over-the-air communication between the PPG and the Push Client and is hence primarily to be used in conjunction with IP bearers. The HTTP variant provides only connection-oriented push. Push content is delivered using the HTTP POST method, implying that the PPG acts as an HTTP client and the Push Client (in the device) as an HTTP server. To avoid confusion the Push Client is therefore referred to as "terminal" in [PushOTA] when OTA-HTTP is discussed.

The [PushOTA] specification defines two methods for establishing an active TCP connection (i.e. a TCP connection to be used for push delivery). The methods are *PPG Originated TCP* (PO-TCP) and *Terminal Originated TCP* (TO-TCP). PO-TCP allows an active TCP connection to be established when the bearer is active (or can be activated by the PPG) and the terminal's IP address is known by the PPG. The TO-TCP method addresses other cases, and is usually used in combination with the Session Initiation Request (SIR) mechanism (see section 5.3.2.2.5).

By using the concept of (long-lived) sessions, OTA-WSP provides a means for the Push Client to convey its capabilities to the PPG. In OTA-HTTP the terminal registers with a PPG to provide similar functionality. This is accomplished by the PPG by sending an HTTP OPTIONS request to the terminal, whereby the terminal includes its Capability and Preference Information (CPI) in the response (optionally using UAProf). A mechanism to avoid the information being sent when it is known by the PPG is provided to improve over-the-air efficiency.

OTA-HTTP also provides a means for identifying and optionally authenticating both the terminal and the PPG. The authentication schemes "basic" and "digest", as specified in [RFC2617], are used to authenticate the PPG to the terminal, while a slightly modified variant is used to authenticate the terminal to the PPG (this since RFC2617 only specifies how an HTTP client, in this case the PPG, is authenticated).

5.3.2.2.3 OTA-SIP

This protocol variant utilises SIP [RFC3261] and MSRP [RFC4975] for communication between the PPG and the Push Client and is hence primarily to be used in conjunction with IP bearers.

In OTA-SIP, the PPG acts as a SIP application server and the Push Client as a SIP client.

OTA-SIP supports both connectionless and connection-oriented push. Connectionless push is supported via the SIP MESSAGE method. Connection-oriented push is supported by using the SIP INVITE method to created a connection between the PPG and Push Client, over which the push content is delivered via MSRP [RFC4975].

5.3.2.2.4 OTA-PTM

This protocol variant utilises point-to-multipoint bearers such as MBMS [MBMS], Cell Broadcast Service [3GPP-CBS] or BCAST [OMA-BCAST-AD] for communication between the PPG and a group of Push Clients. The Push Clients may belong to a specific group targeted by the PI, or a group of Push Clients that are targeted by the service provider in a specific area.

The following sections provide an overview of the various bearer adaptations defined for OTA-PTM.

5.3.2.2.4.1. MBMS Adaptation

The 3GPP Multimedia Broadcast/Multicast Service (MBMS) supports content delivery from one source to multiple recipients using a Broadcast mode and a Multicast mode as described in [23.246].

MBMS enabled clients can receive service announcements either on the MBMS bearers or using interactive methods such as HTTP or using unicast point to point push bearers.

Once the MBMS enabled client receives the service announcement per [26.346], it must initiate the specific service so that it can listen on the indicated MBMS bearer at the indicated time and receive the content as described in [26.346].

MBMS has two delivery methods: streaming and download. Download is used for file delivery and streaming is used for streaming applications such as Video, TV etc…. Streaming content is assumed to be delivered from the BM-SC, and not the PPG. Download of discrete content may be initiated via the PPG.

The PPG may either integrate the functions of a BM-SC or can be connected to a BM-SC using BM-SC provided interfaces, for delivery of the Push OTA content via the MBMS service. As [26.346] does not define an interface for the BM-SC toward the content source (in this case the PPG), the interface between the PPG and BM-SC is implementation-specific.

5.3.2.2.4.2. Cell Broadcast Adaptation

The 3GPP Cell Broadcast Service (CBS) supports broadcast content delivery as described in [3GPP-CBS].

CBS supports simultaneous message delivery to all users of CBS-enabled devices in a specified area. In contrast to SMS as used for connectionless WAP Push (OTA-WSP), CBS is a geographically focused service for point-to-mulipoint content delivery. CBS offers the ability to deliver Push content up to 1230 bytes in length, through concatenation of up to 15 message segments.

To utilize the CBS bearer, the PPG acts as a Cell Broadcast Entity (CBE), and connects to a Cell Broadcast Center (CBC) via the Cell Broadcast Entity Protocol (CBEP). CBEP is a non-standardized protocol exposed by CBC’s towards CBE's, usually based upon HTTP. This protocol is intended to enable delivery of data to specific locations in a GSM/UMTS network, at specified times, and in specific formats.

5.3.2.2.4.3. BCAST Adaptation

OMA BCAST [OMA-BCAST-AD] supports broadcast content delivery to BCAST-enabled devices.

On the network side, Push messages are deliverd via the BCAST File Distribution (FD) Function, across the BCAST-1 reference point defined in [OMA-BCAST-AD]. The Push content is opaque to the BCAST enabler, and uses BCAST FD as a lower-layer transport bearer only.

Future versions of the OMA Push enabler may define use of the BCAST Notification Function for Push message delivery.

On the client side, no standardized reference points are defined by BCAST for application reception of BCAST-delivered content, thus the Push Client interface with the BCAST Client is implementation-specific.

5.3.2.2.5 Session Initiation Application

Regardless which OTA protocol variant is used, it may be necessary for the Push Client to initiate the communication for reasons explained below. The Session Initiation Application (SIA), which is a Push Client-side application, has been specified for this purpose. The SIA listens to Session Initiation Requests (SIR) from the PPG, and responds by activating the appropriate bearer and contacting the desired PPG.

When OTA-WSP is used, it is always the Push Client that takes the initiative to establish the underlying WSP session. An SIR is sent from the PPG to the Push Client when it wishes to create a WSP session for push purposes. Upon reception of the SIR, the Push Client activates the bearer indicated in the SIR and establishes a WSP session towards the indicated PPG over that bearer.

The SIR mechanism is also used in conjunction with OTA-HTTP, in particular when the Push Client's IP address is not known by the PPG, and/or when the PPG cannot activate the desired bearer. In that case the SIR instructs the Push Client to activate a specific bearer and establish an active TCP connection towards the PPG specified in the SIR (using the TO-TCP method).

The SIR mechanism is also used in conjunction with OTA-SIP, in particular when the Push Client is not registered with the SIP/IP Core for push services. In that case the SIR instructs the Push Client to register and contact the PPG as necessary to disclose its capabilities etc.

The SIA may be invoked via OTA-PTM in order to establish a connection-oriented Push bearer, e.g. for delivery of Push content too large to be delivered directly via OTA-PTM, or for point-to-point delivery of Push content based upon user opt-in to a Push content avaibility notification delivered via OTA-PTM.

The SIR is typically sent to the Push Client using connectionless push (provided by OTA-WSP) independent of which protocol variant the Push Client will use when it subsequently contacts the PPG. Attention has been paid to ensure that the SIR is compact enough to fit into a single SMS in the normal case. SMS is available in most current mobile networks, provides a means to use a well-known Push Client address (MSISDN) and provides transport level reliability (i.e. provides good reliability also when connectionless push is used).

5.3.2.2.6 Addressing

Between the PPG and Push Client, two registered ports (secure and non-secure) are used on the Push Client for OTA-WSP connectionless push. When connection-oriented OTA-WSP is used, any WSP session with the push capability set can be used. OTA-HTTP, which is connection-oriented only, uses the concept of active TCP connections, which are dedicated for push specifically. For OTA-SIP, the port is assigned by the SIP/IP Core when the Push Client registers. For PTM-Push, the port can be specified as part of the target address, and is thus shared by the target group. More details about ports, sessions, and active TCP connections can be found in [PushOTA].

5.3.2.2.6.1. OTA Efficiency and Numeric Identifiers

To improve over-the-air efficiency, a numeric identifier may be used instead of a URI. OMA [OMNA] has assigned numbers to well-known user agents, such as the browser on a device [WAE], to avoid the overhead of sending a URI.

If a PPG is requested to push content with an application identifier URI that it recognises as a URI that has a numeric identifier assigned by OMNA, the URI is replaced with the numeric identifier.

5.3.2.3 Push Client-Application Interface

The Push Client-Application Interface (Push-CAI) defines a set of functions and related data formats enabling the high-level capabilities:

- Push application registration with the Push Client, and subsequent deregistration if required. Registration establishes Push Client awareness of the Push application, including any application-specific preferences for Push events, e.g, specific content types which the application is expecting.
- Push Client delivery of Push events to the Push application. When Push events are received, the Push content and relevant metadata, e.g. content type, are delivered to the application.

6. Security Considerations

When implementing Push, security and trust models come into consideration in several areas. These are examples of questions that may arise:

- How can a PI be authenticated?
- What role could the PPG play in the security and trust model?
- What are the access control policies for a PI and pushed content?
- How can a Push Client authenticate something if it has no certificate?

Regardless of these issues, it should be kept in mind that the Push Framework is capable of providing the Push Client with enough information to have a trust model and security policy of its own.

6.1 Authenticating a Push Initiator

It is important that a PI is authenticated in one form or another, depending on the security environments in which the PI and PPG are operating. This is an attempt to list some of the possible solutions.

- **Use of Session-level Certificates (TLS, SSL)**
  If the network between the PI and PPG is not trusted (e.g., the Internet, a very large intranet, etc.), TLS/SSL can be used between the PI and the PPG.

- **HTTP Authentication**
  Even though the most common form of HTTP authentication is the basic authentication (i.e., a user-id/password pair), other forms of HTTP authentication (e.g., digest) might be preferable. The major difference between this approach and the use of TLS/SSL is that the latter is stronger in scalability and confidence, while the former is weaker in these aspects.

- **A Combination of Technologies**
  Technologies could be combined. For example, the PI can establish an anonymous TLS/SSL session with a PPG, whereupon HTTP authentication could be used to authenticate the PI.

- **Shared Secret content type parameters**
  Using a shared secret it may be possible to generate content type parameters to be appended to the push specific content type. These parameters are similar in nature to those used for provisioning bootstrap.

- **Trusted Network**
  In some real world installations, the network between the PI and the PPG is a private network. Therefore, the PI is implicitly trusted in such installations.

6.2 Push Client Delegation of PI Authentication

"Delegation of Authentication" refers to the principle that authentication can be transitive. If a Push Client and a PPG can establish trust, the PPG can authenticate a PI on behalf of the Push Client.

For example, after a Push Client has used the means provided by [WTLS] or [WAPTLS] to authenticate a PPG, the Push Client could look in its list of trusted PPG's. If the PPG is listed as trusted, the Push Client can trust the PPG, and hence also that the PI is correctly identified. Using the methods described in the previous section, a PPG can authenticate a PI with various levels of confidence. If it does, the OTA protocol makes it possible for the PPG to indicate that the PI is authenticated in the message that is delivered to the Push Client.

If the PPG is trusted in this manner then the identity of the PPG may also be cross-checked with an optional filtering 'whitelist' on the Push Client, if the PPG identity is not within the list nor on the provisioned set of proxies on the device any push message received by the Push Client may be ignored.

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
None		

A.2 Draft/Candidate Version 2.3 History

Document Identifier	Date	Sections	Description
Draft Versions: OMA-AD-Push-V2_3	08 Mar 2009		Updated with new OMA AD Template and the agreed CR: OMA-CD-PUSH-2009-0010R02-INP_Push_AD_Updates
	31 Mar 2009		Updated for agreed CR: OMA-CD-PUSH-2009-0021R01-CR_AD-MBMS
	20 April 2009		Updated for agreed CR: OMA-CD-PUSH-2009-0021R03-CR_AD-MBMS
	18 May 2009	9	Updated for agreed CR: OMA-CD-PUSH-2009-0031R01-CR_Push_2.3_AD_Addressing
	16 June 2009		Updated for agreed CRs: OMA-CD-Push2009-037R01-CR_AD_PTM_PPG_changes OMA-CD-PUSH-2009-0038-CR_AD_Push_Architectural_Model OMA-CD-PUSH-2009-0040R01-CR_AD_Push_Architectural_Model_addtl_diagram
	29June 2009		Updated for agreed CRs: OMA-CD-PUSH-2009-0042R01-CR_2.3_AD_edits
	22 July 2009		Updated for agreed CRs: OMA-CD-PUSH-2009-0058R01 OMA-CD-PUSH-2009-0056
	04 Aug 2009		Updated for agreed CRs: OMA-CD-PUSH-2009-0061R01
Candidate Version: OMA-AD-Push-V2_3	13 Oct 2009	All	Status changed to Candidate by TP: OMA-TP-2009-0448-INP_Push_V2_3_AD_for_Candidate_approval
Draft Versions: OMA-AD-Push-V2_3	07 Dec 2009	2.1,4, 5.3.2.2.4, B	Updated for agreed CR: OMA-CD-PUSH-2009-0123
	04 Feb 2010	All	Updated template and editoral update
Candidate Version: OMA-AD-Push-V2_3	16 Mar 2010	All	Status changed to Candidate by TP: OMA-TP-2010-0106-INP_PUSH_V2_3_ERP_for_Candidate_Approval

Appendix B. Example Push Message Delivery (Informative)

Let's assume a PI has submitted a message intended for Push Client Foo, for an application called Bar, to a PPG serving Push Client Foo. In addition, the PI has requested that the message should be delivered in a confirmed manner (implying connection-oriented delivery). The PPG (supporting both OTA-HTTP and OTA-WSP) has not communicated with the Push Client before, so it does not know if it supports OTA-HTTP or OTA- WSP (or both). There's currently no push session or active TCP connection between the PPG and the Push Client called Foo, so either needs to be established.

The PPG sends an SIR to Foo in a connectionless manner using e.g. SMS, indicating that it wants to push some content to application Bar. Since the PPG does not know if the Push Client supports OTA-HTTP or OTA-WSP it includes PPG contact points for both variants in the SIR. This request is sent to the SIA at Foo just like any other push content (i.e. by targeting one of the ports dedicated for connectionless push and including the SIA application identifier). The Push Client receives the content, sees it's for the SIA, and sends it onward. The SIA, on receipt of this request, checks if the target application is installed in Foo and possibly that the user preferences indicate that the target application accepts pushed content. It notes that application Bar is, in fact, installed on this device, so the Push Client acts upon the SIR. Lets assume that the Push Client supports only OTA-WSP, implying that a session should be established towards the PPG.

Now, the owner of this particular device does not want to expose what applications he has installed in the device to anybody (privacy issue). The SIA notes this and sets up a push session with the PPG, indicating that the session accepts content for any application; if the user had been less paranoid, applications for which this session could be used would have been explicitly listed instead.

Once the session has been established, the PPG performs the confirmed push over that session, and the Push Client gets the push content originally submitted by the PI. The Push Client gets the content, sees that it is for application Bar, and passes it to this application. When (and only when) the Bar application takes responsibility for the push content, the push is confirmed all the way back to the PI (if so requested).

Appendix C. Flows (Informative)

C.1 Push Client Registration

Push Clients may be preconfigured to accept Push events from some or all PPG sources. If required by the specific Push-OTA variant, the Push Client will additionally register with the PPG. This registration may be initiated by pre-configuration of the Push Client (e.g. for embedded OMA-standardized enabler client such as a browser or MMS client), or upon registration of an arbitrary Client Application. This flow shows such a generic registration process.

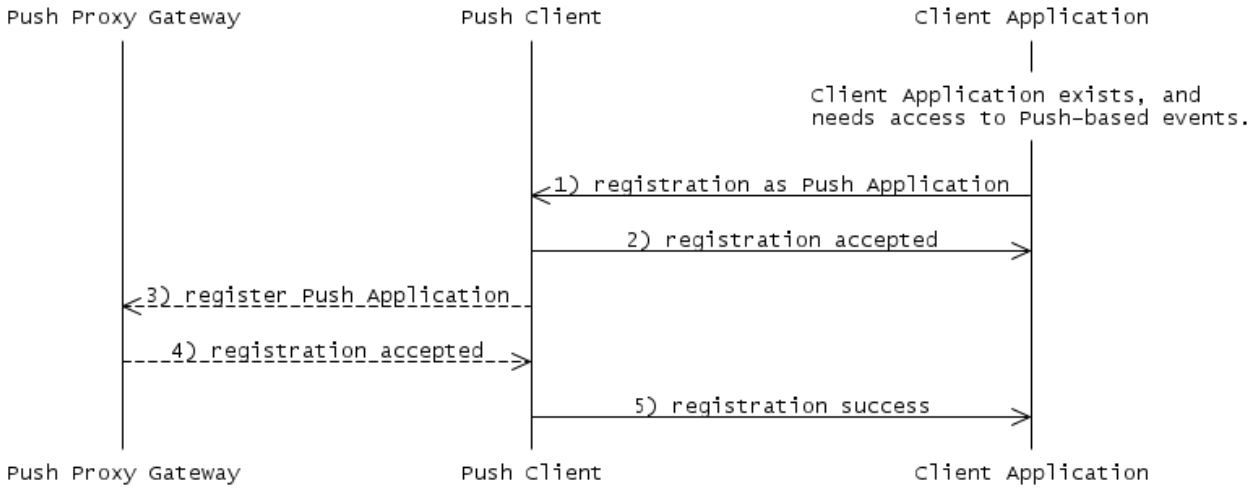

Figure 9 Push Client Registration

1. Via the Push-CAI interface, a Client Application initiates registration or updates an existing registration for Push events provided through a device Push Client. Registration may be requested to whatever PPG's are by default provisioned for the Push Client, or to specific PPG's as required by the Client Application. The Client Application also indicates its Push Application type, and other options as applicable (e.g. requested content types). Note for OMA-standardized enabler clients (e.g. browser, MMS) explicit registration may not be necessary, as the Push Client may designed to automatically register for them.
2. The Push Client confirms acceptance of registration, pending further registration actions that may be necessary.
3. If applicable for the registration request, the Push Client registers with one or more PPG's (as requested or configured) via the Push-OTA interface, indicating the type of Push Application for which service is requested, and other options as applicable (e.g. requested content types). Note that for some Push-OTA variants an explicit registration is not necessary.
4. As applicable to the Push-OTA variant, the PPG accepts the registration request, and may indicate the types of Push service options available, e.g. specific Push Applications. The Push Client may use this information to update the status of Client Application registration requests.
5. The Push Client confirms registration success once all processing is completed successfully.

C.2 Push Client Deregistration

If required by the specific Push-OTA variant, the Push Client will deregister for one or more Push Applications when Push events are no longer necessary/desired for the related Client Applications. This flow shows such a generic deregistration process.

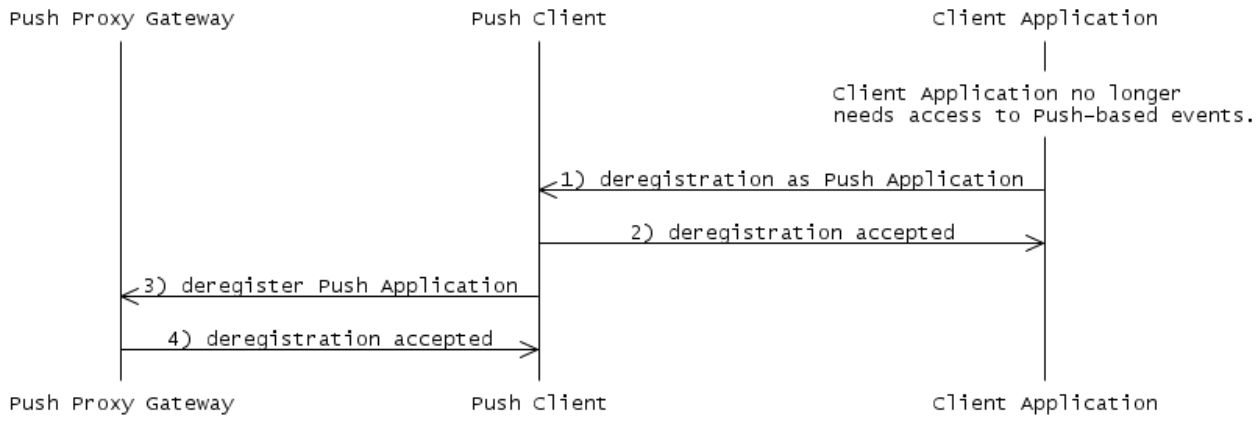

Figure 10 Push Client Deregistration

1. Via the Push-CAI interface, a Client Application initiates deregistration for Push events provided through a device Push Client with which it has en existing registration.
2. The Push Client confirms acceptance of deregistration. Note that further actions may be required by the Push Client in order to complete deregistration with a PPG. For the purpose for Client Application deregistration the request is considered complete at this point.
3. Via the Push-OTA interface, the Push Client deregisters with the PPG's to which it was registered on behalf of the Push Application. Note that for some Push-OTA variants an explicit deregistration is not necessary.
4. As applicable to the Push-OTA variant, the PPG acknowledges the deregistration request.

C.3 Push Message Delivery

This flow shows a Push message delivered as a new or replacement message, to one or more target Push Clients.

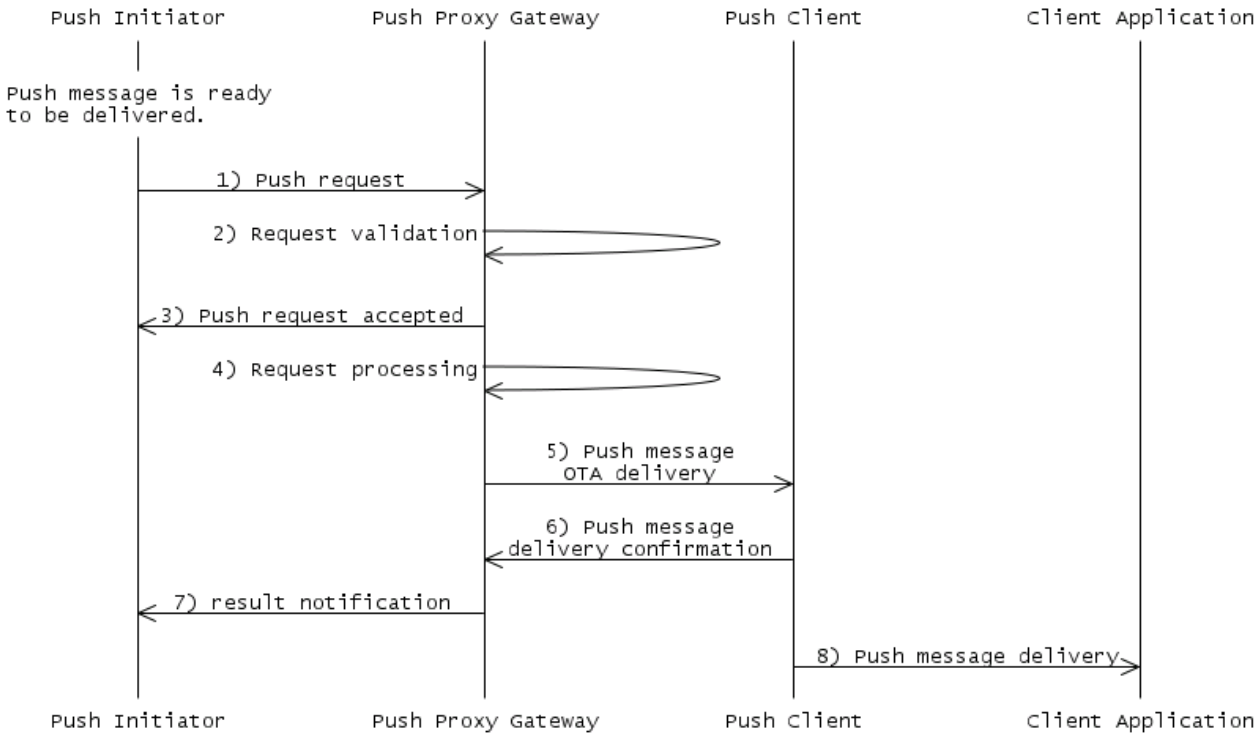

Figure 11 Push Message Delivery

1. Via the PAP interface, a Push Initiator delivers a Push request to the PPG, for delivery as a new message or replacement of a prior message. The Push Initiator indicates the target Push Application and one or more target users to whom the message is to be delivered. The Push Initiator may select various options for message delivery, e.g. bearers, delivery method, delivery scheduling, etc.

2. The PPG validates the request, and may accept or reject the Push request with respect to its validation or other implementation/deployment specific criteria (e.g. policies).

3. Via the PAP interface, the PPG sends a response to the Push request, indicating acceptance. If the PPG had not accepted the Push request, the flow would terminate at this point with a message to the Push Initiator, rejecting the Push Request.

4. For accepted Push requests, the PPG performs Push message processing which may include replacing a Push message previously submitted, queuing, selection of the Push-OTA variant, transformation of the Push message in preparation for OTA transmission, etc.

5. Via the Push-OTA interface, the PPG delivers the Push message to the target Push Client(s). The PPG may deliver the Push request via one or more Push-OTA variants/bearers as necessary to complete delivery to all target Push Clients.

6. If applicable to the Push-OTA variant and options of the Push message, the Push Client confirms delivery of the Push message by the PPG. If the message was not accepted, the Push Client may silently ignore the Push message, or indicate an error in the delivery confirmation.

7. If the Push Initiator had requested a result notification, the PPG will send the result notification to the Push Initiator via the PAP interface after receiving the confirmation message from Push Client, and include a delivery success/failure indication. Note for some Push-OTA variants (e.g. OTA-WSP/SMS), the PPG may

send the result notification earlier in this flow if explicit Push Client delivery confirmation was not possible, and another source of delivery confirmation was used (e.g. delivery confirmation by an SMSC).

8. The Push Client delivers the Push message to the target Client Application, based upon the indicated Push Application. The Push Client will deliver the Push message via the Push-CAI interface if the target Client Application had explicitly registered for Push events via the Push-CAI interface.